(19) Japan Patent Office (JP) (12) Laid-open Patent (A) Gazette (11) Laid-open patent application number Tokkai H. 7-78165

(43) Date of laying-open: March 20, 1995

(51) Int. Cl.[6]	Identification symbols	Internal Patent Office filing numbers	FI	Location for indicating technical field
G06F 17/27	561 F	9379-5H	G06F 15/20	592 F
G10L 3/00		7315-5L		

Request for examination: requested; number of claims: 19; OL (15 pages in all)

(21) Application number: Patent application H. 6-125055

(22) Application date: June 7, 1994

(31) Number of priority claim: P4323241.8

(32) Date of priority claim: July 17, 1993

(33) Country of priority claim: Germany (DE)

(71) Applicant: 390009531
International Business Machines Corporation
Armonk (no number), NY, USA 10504

(72) Inventor: Andreas Arning
3 Elizabethenstrasse, Stuttgart, Germany

(74) Agent: patent attorney Kiyoshi Aida (and 4 others)

(54) [Title of the invention] Method and computer system for detecting error strings in a text

(57) [Abstract]

[Object] To provide a method and computer system for detecting or correcting error strings in a text

[Construction] The present invention relates to a method and computer system for detecting or correcting error strings F_i in a text stored on a computer system. In Step 10 of this method, initially, an error-free string S_i is selected. In Step 11, an error string f_{ij}

that may arise is generated by modifying the error-free string Si. This is useful in the calculating the value of α_{ij} in Step 12. The value of α_{ij} expresses the probability that a possible error string f_{ij} constitutes an actual error string F_i. Consequently, if the value of α_{ij} exceeds a threshold value β, in Step 14, this result is output. Otherwise, Step 15 is executed, after which a further possible error string f_{ij} is generated in Step 11.

10 selection of S_i
11 generation of f_{ij}
12 calculation of α_{ij}
13 to 15: no
13 to 14: yes

[Claims]

[Claim 1] A method for detecting or correcting an error string F_i in a text using a computer system, comprising:

a step of, in order to detect or correct an error string, employing the frequency $H(S_i)$ of the corresponding error-free string S_i in said text, and thereby arranging that the error-free string S_i appears in said text; and

a step of storing said text in said computer system.

[Claim 2] The method as claimed in claim 1 which includes:

(a) a step of modifying an error-free string S_i in accordance with a rule R_i such that a possible error string f_{ij} is generated;

(b) a step of determining the frequency $H(f_{ij})$ of the string f_{ij} in said text;

(c) a step of comparing the frequencies $H(f_{ij})$ and $H(S_i)$; and

(d) a step of deciding whether a possible error string f_{ij} is an actual error string F_i, based on the

comparison of Step (c).

[Claim 3] The method as claimed in claim 2 wherein a step of simulating, by suitable selection of a rule R_i, a source of possible errors that are psychological or technically related to the computer system that is employed, is included in Step (a) of claim 2.

[Claim 4] The method as claimed in claim 2 or 3 wherein, when comparing the frequency $H(f_{ij})$ of a possible error string f_{ij} with the frequency $H(S_i)$ of an error-free string S_i, a step of conducting an evaluation in accordance with a rule for calculating the values $H(f_{ij})$ and $H(S_i)$:

[math 1]

$$\Phi_{ij}(H(f_{ij}),H(S_i))=\alpha_{ij} \qquad (1)$$

is included in Step (c) of claim 2, and a step of then comparing the value α_{ij} with the threshold value β is included in Step (d) of claim 2.

[Claim 5] The method as claimed in claim 4 which includes a step of defining a computer rule Φ as

[math 2]

$$\Phi_{ij}(H(f_{ij}),H(S_i))=(H(S_i))/H(f_{ij}).\Psi)\kappa \qquad (2)$$

when Ψ is a factor and κ is an exponent which is 1 or -1.

[Claim 6] The method as claimed in claim 5 wherein, when $L(S_i)$ is the number of characters in the error-free string S_i and p is an exponent equal to 2 or 3, the factor Ψ is defined as

[math 3] $\Psi=[L(S_i)]^p$

[Claim 7] The method as claimed in claim 4, 5 or 6 wherein, in said computer system, a dictionary-based method is implemented that is employed for the determination of valid strings G_i, and including a step of, for a possible error string f_{ij} having a frequency $H(f_{ij})$ that is greater than 0, using said dictionary-based method to determine whether the string f_{ij} is a valid string, and, if a possible error string f_{ij} is a valid string G_i, modifying the value α_{ij} of the possible

error string f_{ij}.

[Claim 8] The method according to any of claims 4 to 7 which, in said computer system, includes a step of implementing a method for automated learning by assigning to a rule R_j a variable factor δ_j (B) and using that factor to modify the value α_{ij} of a possible error string f_{ij} generated by applying the rule R_j in step (a) of claim 2.

[Claim 9] A method for detecting or correcting error strings F_i in a text, which includes:

(a) a step of detecting the frequencies $H(Z_i)$ of different strings Z_i in the text and defining those strings Z_i having a frequency $H(Z_i)$ exceeding a threshold value γ as error-free strings S_i; and

(b) a step of detecting or correcting an error string F_i associated with an error-free string S_i in accordance with the method of any of claims 1 to 8.

[Claim 10] The method as claimed in claim 9 which includes, in step (b) of claim 2, a step of sorting and storing strings Z_i by their corresponding frequencies $H(Z_i)$ in said computer system and conducting a binary search of said sorted strings Z_i to determine the frequency $H(Z_i)$.

[Claim 11] The method as claimed in claim 10 which includes a step of executing a hashing method of sorting the strings Z_i in accordance with their corresponding frequencies $H(Z_i)$, or using a tree construction constituting a binary tree or linked tries.

[Claim 12] The method according to any one of claims 9 to 11 which includes a step of calculating the corresponding values α_{ij} for various possible error strings f_{ij} of various error-free strings S_i, and automatically replacing those possible error strings f_{ij} that are error strings F_i according to the decision in step (d) of claim 2 by said stored corresponding error-free strings S_i in the text.

[Claim 13] The method as claimed in claim 12 which

includes (a) a step of sorting the various possible error strings f_{ij} according to their corresponding values α_{ij}, and

(b) a step of selecting said criterion such that only those possible error strings f_{ij} that satisfy the criterion of the values α_{ij} are employed in step (d) of claim 2, and thereby using said criterion as a threshold value β.

[Claim 14] A computer system whereby an error string F_i in a text is detected or corrected, thereby causing the corresponding error-free string S_i to appear in said text, comprising:

first storage means for storing said text;

second storage means for storing the frequency $H(S_i)$ of said error-free string S_i; and

processor means using the frequency $H(S_i)$ of the error-free string S_i in detecting or correcting the error string F_i.

[Claim 15] The computer system as claimed in claim 14 which comprises:

third storage means that stores the frequency $H(f_{ij})$ of a possible error string f_{ij} and fourth storage means that stores a rule R_j, and comprising:

means for modifying the error-free string S_i according to the rule R_j whereby said processor can generate a possible error string f_{ij};

means for determining the frequency $H(f_{ij})$ of a possible error string f_{ij};

means for comparing the frequencies $H(S_i)$ and $H(f_{ij})$; and

means for associating a possible error string f_{ij} with the error string F_i based on the output signal from said comparison means.

[Claim 16] The computer system as claimed in claim 15 wherein:

said comparison means provided in said processor means comprises:

calculation means that calculates the value α_{ij} in

accordance with a calculation rule:

[math 4]

$$\Phi_{ij}(H(f_{ij}),H(S_i))=\alpha_{ij} \qquad (1)$$

said output signal transmits the value α_{ij}; and

said associating means comprises means that stores a threshold value β for comparison with the value α_{ij}.

[Claim 17] The computer system according to any one of claims 14 to 16 which comprises:

means for determining the frequency $H(Z_i)$ of different strings Z_i in said text;

fifth storage means for storing the frequency $H(Z_i)$;

means for storing a threshold value γ, and

comparison means for comparing the threshold value γ with a frequency $H(Z_i)$, whereby those strings Z_i having a frequency $H(Z_i)$ exceeding the threshold value γ are defined to be error-free strings S_i.

[Claim 18] A character recognition system comprising an automated optical character recognition system and a computer system according to any one of claims 14 to 17 wherein

said automated optical character recognition system generates raw text for detection or correction of an error string F_i and inputs said raw text into said computer system.

[Claim 19] An automated dictation recording system comprising a speech recognition system and a computer system according to any one of claims 14 to 17 wherein

said speech recognition system generates raw text for detection or correction of an error string F_i and inputs said raw text into said computer system.

[Detailed description of the invention]

[0001]

[Field of industrial application] The invention relates to a method and computer system for detecting or correcting an error string in a text.

[0002]

[Prior art] In known word processing systems,

entered text is stored separate from a dictionary. The dictionary associated with a word processing system is a file that contains a reasonably complete list of known words and if possible their inflected forms, i.e., their conjugations and forms deviating from the standard. When searching for errors in text, each individual word is searched for in the dictionary. If a word is not contained in the dictionary, the word processing system issues an error message and asks the user to check the word. Such systems have been disclosed, for example, in U.S. Pat. Nos. 4,689,678, 4,671,684, and 4,777,617.

[0003] A word processing system has also been disclosed in U.S. Pat. No. 4,674,065, which is based on a statistical N-gram analysis technique. When an incorrect word is detected, the user is offered a list of possible correct alternatives to select from.

[0004] An overview of known techniques for automated correction of words in a text is provided by the publication "Techniques for Automatically Correcting Words in Text," by Caron Kukich, ACM Computing Surveys, Volume 24, No. 4, December 1992.

[0005] The known methods for error detection and correction share the characteristic that a dictionary separate from the text is used as the standard for comparison. The known systems thus require a relatively large amount of memory for storing the dictionary, and this memory is thus not available to other applications.

[0006] A further disadvantage of using a dictionary is that, in general, the dictionary itself contains some errors and thus cannot be relied upon as a standard. After all, the dictionary itself cannot be checked for errors by the word processing system, since a dictionary is considered to be the most reliable standard. Moreover, the dictionary must be continually updated, allowing additional errors to creep in. The use of known word processing systems is practically

unsuitable for checking multilingual texts, since all "foreign words" not present in the dictionary will be flagged as errors. The same also holds true for monolingual texts which employ unusual words or newly-coined words, as well as for computer code or texts that contain phonetic information or formatting controls. In these cases, known word processing systems may flag a large number of correct strings as incorrect, since the strings do not occur in the dictionary. This problem is especially evident when the text being checked includes abbreviations or formulas or contains proper names that are not stored in the dictionary.

[0007]

[Problem that the invention is intended to solve] The invention is thus based on the object of providing an improved method and computer system for detecting or correcting an error string in a text.

[0008]

[Means for solving the problem] The object of the invention is achieved by a method for detecting or correcting an error string F_i in a text using a computer system, comprising: a step of, in order to detect or correct an error string, employing the frequency $H(S_i)$ of the corresponding error-free string S_i in said text, and thereby arranging that the error-free string S_i appears in said text; and a step of storing said text in said computer system.

[0009] Also, the object of the invention is achieved by a computer system, in particular a word processing system, wherein an error string F_i in a text is detected or corrected, thereby causing the corresponding error-free string S_i to appear in said text, comprising: first storage means for storing said text, second storage means for storing the frequency $H(S_i)$ of said error-free string S_i, and processor means using the frequency $H(S_i)$ of the error-free string S_i in detecting or correcting the error string F_i.

[0010] With the invention, the storing of a dictionary is not required, so the disadvantages previously described for the prior art systems are largely eliminated. In contrast to known word processing systems, according to the invention, the text is not checked with respect to a dictionary but is rather itself subjected to a statistical analysis which serves as the basis for error detection. Here, external dictionaries are not required. The frequency of the error-free string, given by the user, in the text forms the basis for detecting error variants of the string. The frequency of the error-free string serves as a measure for the probability that a possible error string in the text is an actual error string corresponding to the error-free string. The error string identified in this manner, if it occurs more than once in the text, can then be replaced automatically by the corresponding error-free string throughout the entire text.

[0011] In one embodiment of the invention, the error-free string specified by the user and occurring in the text is modified according to at least one rule, so that one or more possible error strings are generated. In deciding whether a possible error string actually corresponds to an error-free string given by the user, the frequency of the possible error string in the text is determined. The frequencies of the error-free string and the possible error string are compared, and this comparison forms the basis for deciding whether the possible error string is an actual error string. The comparison of the frequencies uses the fact that a word occurring frequently in a text has, with high probability, been entered incorrectly at least once. Thus, the larger the ratio of the frequency of the error-free string to the frequency of the possible error string, the higher the probability that the possible error string is an actual error string.

[0012] To increase the effectiveness of this search

for error strings in the text, in accordance with a preferred embodiment, the rule or rules used to modify the error-free string are selected such that psychological errors or error sources related to the computer system, in particular to its keyboard, or both of these, are simulated. A keyboard-related error, for example, is pressing a key adjacent to the desired character. If, for example, due to the keyboard used, the character "b" frequently occurs in place of its neighbor "v", this can be allowed in a corresponding rule. By applying the corresponding rule, a "v" occurring in the error-free string is replaced with "b", so that, from the error-free string, a possible error string is generated. This also occurs in the text with high probability. For any one single error-free string, this procedure can be repeated using different rules to simulate different possible errors.

[0013] The probability that applying a specific rule will generate a possible error string that actually occurs in the text can, depending on the rule, vary with the user, the computer system used, or both. This probability can be subject to time-related variations, for example, because the user has learned to avoid certain kinds of errors, because a new user takes over and tends to make other kinds of errors, or because the computer system used is replaced with another, having another keyboard. This can be taken into consideration using a method of automated learning that registers the success probabilities of the rules employed. If the automated learning process shows that a rule often leads to detecting an error string, this rule will be given preference and weighted with a factor. An initialization of these factors can also be determined using a training sequence.

[0014] In accordance with a further preferred embodiment, the entire text is automatically checked. In this case, the frequencies of all unique strings in the text are first determined. The strings whose

frequencies are higher than a specified threshold value are defined as error-free strings, since a string occurring very often in a text has a high probability of being correct. The error-free strings so defined, or their frequencies, then serve as the basis for error detection.

[0015] In accordance with a further preferred embodiment, the invention relates to a character recognition system comprising a system for automated optical character recognition. The system for automated optical character recognition can, for example, be used to enter a printed text into a computer system. In this case, the raw text input to the computer system for the automated optical character recognition process is not error-free. The generation of errors can result from the fact that the printed text contains errors or that the system for automated optical character recognition does not function without errors. The raw text entered into the computer system is checked by the computer system for errors in accordance with the invention, so that in particular deficiencies in the system for automated optical character recognition can be corrected over a wide range. A method based on an N-gram technique for supporting an apparatus for character recognition is disclosed in U.S. Pat. No. 4,058,795.

[0016] In accordance with a further preferred embodiment, the invention relates to a system for automated recording of dictation, comprising a speech recognition system. Such speech recognition systems have been disclosed, for example, in U.S. Pat. Nos. 4,783,803; 4,741,036; 4,718,094; and 4,164,025.

[0017] The speech recognition system generates a raw text, generally exhibiting errors, which is entered into a computer system. The error detection or correction provided by the invention is applied using such a computer system.

[0018] In accordance with a further preferred

embodiment, the invention relates to a storage medium suited for use in a programmable computer system. Through a physical or chemical process, a program is recorded on the storage medium for carrying out the inventive method. Through this physical or chemical process, the storage medium acquires the characteristic of being able to interact with the programmable computer system such that said computer system, or a conventional computer system programmable for general purposes, is transformed into a computer system according to the invention.

[0019]

[Embodiments] The block diagram shown in FIG. 1 refers, for example, to a word processing system according to the present invention, into which a text to be checked has already been entered. In step 10, the user can select an error-free string S_i occurring in the text. The object of the inventive method is then to detect at least one error string F_i in the text corresponding to the selected error-free string S_i, i.e., representing, for example, a typographical error when compared with the error-free string S_i.

[0020] Next, in step 11, a possible error string f_{ij} is generated. The possible error string f_{ij} is produced from the error-free string S_i by applying rule R_j. In step 11, by use of rule R_j if possible on different letters or letter positions, multiple possible error strings f_{ij} are generated from the error-free string S_i.

[0021] In step 12, a value α_{ij} is calculated as the comparison of the frequency $H(S_i)$ of the error-free string S_j and the frequency $H(f_{ij})$ of the possible error string f_{ij}.

[0022] In step 13, the value α_{ij} calculated in step 12 is compared with a threshold value β. If $\alpha_{ij}>\beta$, the search result is declared in step 14 to be that the possible error string f_{ij} is equal to an actual error string F_i. This result can be used for automated correction of all strings F_i occurring in the text.

Prior to this correction, the results so obtained can be presented to the user for verification. In this case, the automated correction is carried out only if the user agrees with the proposed result.

[0023] If the condition $\alpha_{ij}>\beta$ is not true, the index j in step 14 is incremented by 1. The result of this is that in the next step 11 for generating another possible error string, another rule R_j+1 is applied. The additional possible error string $f_i+1,j+1$ so generated represents an additional candidate, which could correspond to an error string F_i. This determination is again made in the subsequent steps 12 and 13, and, if applicable, the result is declared in step 14.

[0024] According to the flow diagram in FIG. 1, the method is then terminated as soon as an error string F_i is determined in step 14. It can also happen, however, that, in this case too, additional possible error strings f_{ij} can be accidentally formed by applying other rules R_j. This corresponds to the steps 15 and 11 described above. In this way, still more error strings F_i can be found that, for example, have arisen through other entry errors with respect to the error-free string S_i selected in step 10.

[0025] In this case, it is also possible that initially in several sequential steps 14, different error strings F_i are defined as results of the search and these error strings are presented to the user sorted by the corresponding α_{ij} values. Since the α_{ij} values represent a measure of the probability that a possible error string f_{ij} is an actual error string F_i occurring in the text, the results are therefore shown to the user sorted by their probability.

[0026] In contrast to known dictionary or N-gram based systems, the basis for the error detection is not externally stored data - such as in the form of a separately stored dictionary - but rather the text itself being checked. In accordance with the invention, the otherwise externally stored data is derived from

the text being checked by determining the frequency $H(S_i)$. If the frequency $H(S_i)$ assumes high values, the invention leads to the conclusion that a possible error string f_{ij} occurring seldom in the text represents an actual error string F_i. In this case, externally stored data and the attendant expenditure are not required.

[0027] The rules R_j employed in step 11 for generating the possible error strings f_{ij} are preferably selected such that psychological errors and/or other error sources related to the computer system, in particular to its keyboard, are simulated. Psychological errors are those, for example, that are difficult to find when copy editing, such as errors in particularly long words. A keyboard-related error is, for example, one caused by inadvertent bouncing, producing a double letter. Inadvertent multiple entry or omission of a character at the keyboard can also occur if the keyboard exhibits a poorly defined action point.

[0028] The calculation in step 12 of the value α_{ij} can be performed based on the computing rule of equation (1) below.

[0029]

[math 5]

$$\Phi_{ij}(H(f_{ij}),H(S_i))=\alpha_{ij} \qquad (1)$$

This computing rule can preferably have the form

[0030]

[math 6]

$$\Phi_{ij}(H(f_{ij}),H(S_i))=(H(S_i))/H(f_{ij}).\Psi)\kappa \qquad (2)$$

where Φ_{ij} is a function dependent on the frequency $H(f_{ij})$ and the frequency $H(S_i)$, the value Ψ is a factor, and the value κ represents an exponent.

[0031] The factor Ψ can be calculated according to the equation (3) below.

[0032]

[math 7]

$$\Psi=[L(S_i)]^p$$

where, using the function L, the length of the

string S_i, or in other words, the number of characters in string S_i is determined. The value p represents an exponent that is preferably quadratic or cubic.

[0033] The following formula is the quotient contained in equation (2).

[0034]

[math 8] $H(S_j)/H(f_{ij})$

This quotient is the key element in computing the value α_{ij}. The reason for this is that this quotient increases with increasing frequency of the error-free string S_i and decreasing frequency of the possible error string f_{ij} in the text. This quotient is based on the experience that a string occurring with high frequency in a text has a high probability of being correct, and that furthermore the probability that the usable string also occurs in the text at least once with an error - e.g., due to an entry error - increases with the frequency of the error-free string in the text. Using this correction factor Ψ can also take into account that with increasing string length, the probability that the string contains an error increases, in particular too because errors in long strings are generally not easily recognized by the user. Furthermore, the factor Ψ takes into consideration that with increasing word length the probability decreases that a modification of the error-free string S_i using a rule R_j will lead to another error-free string S_i occurring in the text. This has particularly strong influence on the calculation of the value α_{ij}; in this case a value such as 2 or 3 is chosen for the exponent p. The value k in the first embodiment of FIG. 1 has the value 1. If this value is chosen as -1, only the condition $\alpha_{ij}>\beta$ in step 13 need be replaced by $\alpha_{ij}<\beta$. For simplified representation, only the case k=1 will hereafter be considered, without loss of generality.

[0035] The value α_{ij} calculated using equation (1) thus increases with the probability that a possible error string f_{ij} is an actual error string F_i. In step

13, a check is therefore made whether the result based on comparing the frequencies $H(S_i)$ and $H(f_{ij})$ provides a sufficient measure of safety for the definition of a result in step 14. The choice of the corresponding threshold value β thereby depends on the requirements of the user: a high threshold value means that the result determined in step 14 is almost certainly correct, while possible error strings f_{ij} that also lead to a correct result are discarded in step 13. The opposite is true if a low value is chosen for the threshold value β.

[0036] The following tables Table 1 to Table 19 show several examples of possible rules R_j. Also, for each rule an example is given with an error-free string S_i, the corresponding possible error string f_{ij}, and the related value α_{ij}. Following the strings S_i and f_{ij}, their corresponding frequencies in the examined text are given. The text is from the sports sections of the "Frankfurter Rundschau" newspaper for 1988.

[0037]

[Table 1]

Rule R_1 : Transposition of two successive letters.

[0038] Example:

f_{11}="Olmypischen"(1)

S_i = “Olympischen” (875)

α_{11}=1164625

[0039]

[Table 2]

Rule R_2 : Omission of a letter occurring at least twice.

Example:

f_{22}="Präsidumssitzung"(1)

S.sub.2="Präsidiumssitzung"(7)

α_{22}=40824

[0040]

[Table 3]

Rule R_3 : Omission of a letter occurring at most once.

[0041] Example:

f_{33}="Diziplinen"(1)

S_3="Disziplinen"(89)

α_{33}=118549

[0042]

[Table 4]

Rule R_4 : Doubling of a letter.

[0043] Example:

f_{44}="Baskettball"(2)

S_4="Basketball"(179)

α_{44}=89500

[0044]

[Table 5]

Rule R_5: Replacement of a letter.

[0045] Example:

f_{55}="Golopprennbahn"(1)

S_5="Galopprennbahn"(34)

α_{55}=93296

[0046]

[Table 6]

Rule R_6 : Insertion of a letter not previously occurring in the word.

[0047] Example:

f_{66}="Wiederanspfiff"(1)

S_6="Wiederanpfiff"(47)

α_{66}=103259

[0048]

[Table 7]

Rule R_7 : Insertion of a letter previously occurring in the word.

Example:

f_{77}="Ablöseseumme"(1)

S_7="Ablösesumme"(91)

α_{77}=157248

[0049]

[Table 8]

Rule R_8 : Incorrect doubling of a letter, here: left-hand neighbor.

[0050] Example:

f_{88}="Spvvg"(4)

S_8="Spvgg"(142)

α_{88}=4435

[0051]

[Table 9]

Rule R_9 : Incorrect doubling of a letter in a word, here: right-hand neighbor.

[0052] Example:

f_{99} ="Sperrwerfen"(1)

S_9 ="Speerwerfen"(19)

α_{99}=25289

[0053]

[Table 10]

Rule R_{10} : instead of the desired letter, right-hand neighbor was pressed.

[0054] Example:

f_{1010}="erfolgteich"(1)

S_{10}="erfolgreich"(290)

α_{1010}=385990

[0055]

[Table 11]

Rule R_{11} : in addition to the desired letter, right-hand neighbor was pressed; insertion before intended letter.

[0056] Example:

f_{1111}="Cjhristian"(1)

S_{11}="Christian"(175)

α_{1111}=127575

[0057]

[Table 12]

Rule R_{12} : in addition to the desired letter, right-hand neighbor was pressed; insertion after intended letter.

[0058] Example:

f_{1212}="Verletzunmg"(1)

S_{12}="Verletzung"(153)

α_{1212}=153000

[0059]

[Table 13]

Rule R_{13} : instead of the desired letter, left-hand neighbor was pressed.

[0060] Example:

f_{1313}="Problene"(1)

S_{13}="Probleme"(290)

α_{1313}=148480

[0061]

[Table 14]

Rule R_{14} : in addition to the desired letter, left-hand neighbor was pressed; insertion before intended letter.

Example:

f_{1414}="Hoffnungsträgwer"(1)

S_{14}="Hoffnungsträger"(18)

α_{1414}=73728

[0062]

[Table 15]

Rule R_{15} : in addition to desired letter, left-hand neighbor was pressed; insertion after intended letter.

[0063] Example:

f_{1515}="Qualkifikation"(1)

S_{15}="Qualifikation"(255)

α_{1515}=560235

[0064]

[Table 16]

Rule R_{16} : Capitalization error on first letter.

[0065] Example:

f_{1616}="olympiastadion"(1)

S_{16}="Olympiastadion"(5)

α_{1616}=13720

[0066]

[Table 17]

Rule R_{17} : Capitalization error on second letter.

[0067] Example:

f_{1717}="SChwalbach"(1)

S_{17}="Schwalbach"(38)

α_{1717}=38000

[0068]

[Table 18]

Rule R_{18} : Omission of a double letter, leaving only single letter.

[0069] Example:

f_{1818}="Etapensieger"(1)

S_{18}="Etappensieger"(37)

α_{1818}=81289

[0070]

[Table 19]

Rule R_{19} : Doubling of a doubled letter, thus tripling it.

[0071] Example:

f_{1919}="UdSSSR"(1)

S_{19}="UdSSR"(740)

α_{1919}=92500

The rules R_j are optimally selected when essentially only those variants best corresponding to the observed error types are generated in step 11. In this respect, the following rules have proven themselves: rule R_1 (transposition of two successive letters: from "abcba", "bacba", "acbba", "abbca" and "abcab"), rule R_2 (omission of one letter of letters occurring at least twice, i.e., excepting individual letters, which occur only once: for example, from "abcba": "bcba", "acba", "abca", and "abcb"); and rule R_7 (insertion of individual letters, that have previously appeared: for example, from "abc": "aabc", "abac", "abca", "babc", "abbc", "abcb", "cabc", "acbc", "abcc", but not "abdc" or the like).

[0072] Rule R_2 serves primarily to simulate a possible psychological error source. Omissions of letters occur very easily during manual entry but are more difficult to find during copy editing if the omitted letter occurs again in the string - because it is then not "missed" so much.

[0073] On the other hand, rules R_{10} to R_{15} serve to

simulate technical deficiencies of the entry method employed - in this case a keyboard. The technical deficiency of the keyboard makes itself evident in this example in the ergonomically unfavorable formation of the keys, so that adjacent keys are frequently pressed by mistake.

[0074] A further possible rule is the replacement of optically similar letters in the error-free string, e.g., replacement of "c" by "e". In a word processing system according to the invention, use of this rule can simulate error sources arising from technical deficiencies (such as insufficient resolution) of the screen used to display the text. In a character recognition system in accordance with the invention, this and other rules can simulate technical deficiencies of the system for automated optical character recognition, since optically similar letters are often not correctly recognized by such systems. In the same way, in a system for automated recording of dictation, in accordance with the invention, deficiencies in the associated speech recognition system can be simulated. Applying the corresponding rules, phonetically similar letters are transposed, e.g., "m" with "n", since speech recognition systems often produce such errors. Of course, the rules mentioned can apply not only to words but also to strings of any composition.

[0075] In calculating the value α_{ij} in step 12, a dictionary-based method can be used in addition. The possible error string f_{ij} is then additionally checked using the dictionary-based method. If the string f_{ij} is contained in the dictionary, i.e., if it is a valid string G_i, this would initially indicate that the possible error string f_{ij} is not an error. However, this is in no way certain, since an error in the corresponding error-free string S_i can by chance also lead to a valid string Gi, i.e., the possible error string f_{ij} can occur in the dictionary as a valid string

G_i in addition to being an error string F_i. Of course, as noted, a certain probability exists that a possible error string f_{ij} occurring as a valid string G_i in the dictionary is not an actual error string F_i. This can be taken into account in the calculation of α_{ij} by modifying the value α_{ij} from equation (2) if the string f_{ij} is a valid string G_i. The modification can be done by multiplying the value from equation (2) by a factor between 0 and 1. The factor 0 here signifies that a valid string G_i is defined unconditionally as error-free. In this case, however, an advantageous characteristic of the inventive method would be lost, namely consideration of the context. Using the inventive method, the word "director" in a manual for data processing was determined to be a possible error variant of the word "directory", although the word "director" is valid. Taking context into account is implicit in the inventive method, since the frequencies $H(S_i)$ and $H(f_{ij})$ are compared with each other. The factor is advantageously chosen to be significantly greater than 0.

[0076] The calculation of the value α_{ij} in step 12 can be further influenced by a method for automated learning. The method for automated learning assigns a factor δ_j (B) to an applied rule R_{ij}. The factor δ_j (B) is a variable and can be influenced on the one hand by the user and on the other hand by the type of hardware used. If the application of a rule R_j leads by the aforementioned average frequency to finding an error in the text, the method for automated learning assigns the rule R_j a corresponding factor δ_j (B) greater than 1. In the opposite case, the method for automated learning assigns the rule a factor less than 1. The value α_{ij} determined in step 12 from equation (2) is thus additionally multiplied by the factor δ_j (B) associated with the applied rule R_{ij}, so that the different success probabilities of the rules R_j are considered in the calculation of α_{ij}. The rules R_j can be sorted according

to their factors δ_j (B) such that the rules R_j with the highest probability of success, to which a relatively large factor δ_j (B) is assigned, are applied first in step 11. If the inventive method is conducted fully automatically, i.e., without displaying the detected errors as suggestions to the user, the definition in step 14 is the key for the method of automated learning. If a suggestion is given to the user, the user's acceptance of a string proposed as being in error is the key for the method of automated learning and thus for determining the factors δ_j (B). The method for automated learning can, for example, be implemented with a neural network, probably in conjunction with an expert system.

[0077] By using a system for automated learning, a user- and/or hardware-specific calibration can be implemented. For example, the transposition of "y" and "z", such as in "Szstem", can be expected only with those users who continually switch between German and American keyboards, but not with authors of newspaper copy, who generally work with only one type of keyboard. Since there are also corresponding word pairs which do not represent errors, for example "Holy" and "Holz", it is useful to consider such transpositions as possible errors only if they are reasonable for the application area. A hardware-related type of error that can be allowed through the method of automated learning is, for example, the inadvertent simultaneous depression of two adjacent keys on the keyboard, such as in "Sysrtem". The probability of this type of error will depend on the keyboard used - in particular its action point and any generation of an acoustical signal when pressing a key. Furthermore, the method of automated learning can also allow for the use, prior to the inventive method, of other spell-check methods which detect certain error types with difficulty. Those rules R_j which simulate these error types then are assigned a particularly heavy weighting via the factor

δ_j (B).

[0078] The user- and/or hardware-specific calibration can also be obtained by direct entry of the user- or hardware-specific weighting factors δ_j (B). The factors δ_j (B) associated with a specific user, specific hardware, or a specific combination of user and hardware, can then be stored in separate data sets. If the user or hardware changes, the current set of factors δ_j (B) is replaced by the set of factors δ_j (B) associated with the new user or the new hardware, so that the latter set becomes the current one. The current set of factors δ_j (B) serves to weight the values obtained from equation (2) in step 12. The value α_{ij} is thus obtained by multiplying the value obtained from equation (2) with the factor δ_j (B) associated with the applied rule R_j. The current set of factors δ_j (B) thus obtained can also serve as a set of initial values for the factors δ_j (B) for the method of automated learning, so that the method can start off with user- or hardware-specific weighting factors δ_j (B), which can then be further optimized automatically. If the user or hardware changes, the optimized set of factors δ_j (B) can be stored for later use as initial values.

[0079] In addition, it is beneficial to provide an exception table, in which frequent word pairs such as form/from or three/there are stored. Suitable names can also be stored in this table, e.g., Helmut/Hellmut or Hausmann/Haussmann. These could also arise from typographical errors. These words are thus not regarded as possible error strings in step 12. For a possible error string f_{ij} generated in step 11, a check is made whether this string f_{ij} is present in the exception table. If so, the next step executed will be 15 rather than 12.

[0080] FIG. 2 shows the flow diagram of a second preferred embodiment of the invention. In step 20, the frequency $H(Z_i)$ of each string Z_i occurring in the text is first determined. In this case, each unbroken

sequence of letters or any other characters, depending on the application, can be defined as a string Z_i.

[0081] In step 21, the occurring strings Z_i and their corresponding frequencies $H(Z_i)$ are stored pairwise in a table. In step 22, the condition $H(Z_i)>\gamma$ is tested. The value γ is a threshold value for the frequency $H(Z_i)$, above which the corresponding string Z_i is defined to be an error-free string S_i. If, therefore, the frequency $H(Z_i)$ of a specific string Z_i exceeds the threshold value γ, this specific string Z_i is defined as an error-free string S_i. The basis for this is that a string occurring relatively often in a text is with high probability an error-free string or a correctly spelled word of the language that can be used.

[0082] If the condition $H(Z_i)>\gamma$ in step 22 is not satisfied, the next step executed is 23, in which the index i is incremented by one. In the subsequent step 22, the condition $H(Z_i+1)>\gamma$ is tested for another string.

[0083] If the condition $H(Z_i)>\gamma$ in step 22 is satisfied by a string Z_i, step 24 is executed next. In step 24, the corresponding string Z_i is defined as an error-free string S_i. The subsequent steps 11, 12, 13, 14, 15 correspond to the steps of the first embodiment discussed with reference to FIG. 1. Step 24 thereby performs the function of step 10 in the first embodiment, namely the selection of a specific error-free string S_i. All possible variations previously discussed with respect to the first embodiment are also possible in the second embodiment.

[0084] After completing the search for error strings F_i of the string S_i defined as error-free in step 14, the condition $i=i_{max}$ is tested in step 25. If index i has reached the maximum value i_{max}, all strings Z_i occurring in the text have been examined, so that the process is terminated in step 27.

[0085] If the condition $i=i_{max}$ is not yet satisfied, the index i is incremented by 1 in step 26, and in step

22 the condition $H(Z_i+1)>\gamma$ is again checked for another string Z_i.

[0086] Step 12 for computing the value α_{ij} can advantageously be carried out by obtaining the frequency $H(f_{ij})$ from the table stored in step 21, so that the calculation is accelerated. If a possible error string f_{ij} is not present in the table, its frequency is 0. In this case, step 15 can be executed without further evaluation, so that another rule R_j can be applied to generate another possible error string.

[0087] The result obtained in step 14 can be used for automated correction, as previously discussed with reference to FIG. 1. It can be beneficial, however, to store all results obtained in step 14 and, after executing step 27, sort them by the corresponding values of α_{ij}. The user is then presented with a result list from which the user can accept or reject individual results for automated correction. Since the list is sorted by the values α_{ij}, the most reliable results are shown first. If the threshold value β was selected relatively large, however, this procedure is not necessary, since in general all the results actually obtained in step 14 can be used, so that an automated correction can take place immediately without user intervention.

[0088] To limit execution time of the process, e.g., because only a certain amount of computing time is available, the method can be terminated prematurely if a defined number of errors have already been found or a certain portion of computing time has been expended. To accelerate the process, the generation of possible error strings f_{ij} can be controlled such that all rules R_j are applied only if the frequency $H(S_i)$ for the error-free string S_i associated with a possible error string f_{ij} is high. In general, this expenditure will be worthwhile only if the frequency $H(S_i)$ is very high. A high $H(S_i)$ frequency implies a large statistical sample set, so that the reliability of the result in

step 14 increases. In the case of lower $H(S_i)$ frequencies, the set of rules R_j used for detecting an error string F_i associated with an error-free string S_i can be limited accordingly, so that steps 11 to 15 are executed faster overall.

[0089] If prior to executing step 22 the table generated in step 21 is sorted, e.g., in alphabetical order, further acceleration results. The search of the table of possible error strings f_{ij} for calculating the value α_{ij} in step 12 can then be carried out as a binary search. The binary search method is well-known, e.g., from Donald E. Knuth, "The Art of Computer Programming," Vol. 3, Section 6.4.1, Algorithm B, Addison-Wesley Publishing Company, 1973.

[0090] In FIG. 3, a further possibility for storing the table generated in step 21 is shown. The tree structure depicted in FIG. 3 is generally described as a "linked trie", (see references such as Franklin Mark Liang, "Word Hy-phen-a-tion by Com-put-er", Department of Computer Science, Stanford University, August 1983, pp. 11ff. and the references cited therein; de la Briandais, Rene, "File searching using variable length keys," Proc. Western Joint Computer Conf. 15, 1959, pp. 295-298; and Fredkin, Edward, "Trie memory," CACM 3, September 1960, pp. 490-500.) In this example, the tree includes nodes 30, each node 30 having entries 31 through 34. Entry 31 contains a letter or symbol, entry 32 contains the frequency $H(Z_i)$ of the corresponding string Z_i, entry 33 is a pointer to a child - if present - of node 30, and entry 34 is a pointer to a sibling - if present - of node 30. Entry 32 in a node 30 is non-zero if the string from the highest level of the tree to the node 30 occurs in the text. An example is shown in FIG. 3 on the basis of a text that contains only the words "Festung", "Feuer", "Rauch", "Frieden", and "Fest", whereby the word "Feuer" occurs twice and the word "Fest" occurs three times in the text. The remaining words each occur only once in the text.

[0091] This type of storing of the table in step 21 has the advantage of requiring less storage space and providing added acceleration to the process. The structure of the "linked trie" can be generated in parallel with determining the individual strings Z_i and their frequencies, so that subsequent sorting is unnecessary. The applicable algorithm has been specified by Knuth (reference: Donald E. Knuth, "The Art of Computer Programming," Addison-Wesley Publishing Company, 1973, Section 6.2.2, pp. 422 ff., in particular Algorithm T).

[0092] FIG. 4 shows an embodiment of a computer system in accordance with the invention. The computer system comprises storage means 1 for storing the text to be checked; storage means 12 for storing the frequencies $H(Z_i)$, or, in other words, for storing the table or tree structure established in step 21 (see FIG. 2 and FIG. 3); storage means 4 for storing rules R_j used in step 11 for generating the possible error strings f_{ij} (see FIG. 1 and FIG. 2); and processor means 2 for process control. The processor means 2 employs the frequency $H(S_i)$ of the error string F_i for detecting the error string F_i. The storage means 1, 4, 12 and the processor means 2 are interconnected via a bus 16, so that the processor means can access the respective storage means. In this embodiment, the processor means 2 comprises storage means 3 for storing a frequency $H(f_{ij})$ needed to compute the value α_{ij} in step 12; means 5 for modifying an error-free string S_i in accordance with a rule R_j, whereby a possible error string f_{ij} can be generated according to step 11; means 6 for determining the frequency $H(f_{ij})$; means 7 for comparing the frequencies $H(f_{ij})$ and $H(S_i)$; means 8 for associating the possible error string f_{ij} with the error string F_i; means 11 for determining the frequency $H(Z_i)$ of various strings Z_i in the text; and comparison means 13 for comparing the threshold value γ with the frequency $H(Z_i)$. The means 3, 5, 6, 7, 8, 11, 13 are

interconnected via a processor-internal bus 16. The means 3, 5, 6, 7, 8, 11, and 13 contained in the processor means 2, as well as bus 16, need not be discrete electronic components but can rather be generated via appropriate programming of processor means 2. Such a program suitable for implementing the inventive method will interact with the control program of the computer system in a well-known manner such that the computer system assumes the configuration shown in FIG. 4.

[0093] The means 6 for determining the frequency $H(f_{ij})$ interacts via the bus 16 with means 12 such that the desired frequency $H(f_{ij})$ can be derived from means 12, if this frequency is stored there. If there is no entry for the possible error string f_{ij} in the table stored in means 12, the frequency $H(f_{ij})$ is zero. The determination of the frequency is needed to calculate the value α_{ij} in step 12.

[0094] The means 7 for comparing the frequencies $H(S_i)$ and $H(f_{ij})$ comprises computing means 9 for computing the value α_{ij} in accordance with the following computation rule.

[0095]

[math 9]

$$\Phi_{ij}(H(f_{ij}),H(S_i))=\alpha_{ij} \qquad (1)$$

This corresponds to the comparison of the frequencies $H(S_i)$ and $H(f_{ij})$ carried out in step 12 in computing the value α_{ij}.

[0096]

The means 8 for associating the possible error string f_{ij} to the error string F_i comprises means 10 for storing the threshold value β for a comparison with the value α_{ij}. The value α_{ij} determined for comparison by means 7 is transferred via bus 16 to associating means 8. Associating means 8 processes the value α_{ij} in accordance with steps 13 and 14.

[0097] The means 11 for determining the frequency $H(Z_i)$ interact with the means 1 to identify individual

strings Z_i in the text and to calculate the corresponding frequencies $H(Z_i)$, in accordance with step 20.

[0098] The comparison means 13 includes means 14 for storing the threshold value γ. The comparison means 13 interact with means 11 to define those strings Z_i as error-free strings S_i whose frequency $H(Z_i)$ exceeds the threshold value γ. Using appropriate control by a program 17, the computer system in accordance with the invention can thereby carry out the procedure of FIG. 1 and FIG. 2. The program can be stored in the means 17 for program control, whereby the means 17 for program control interacts with the processor means 2 via bus 16.

[0099] Using the computer system in accordance with the invention, the sports sections of the "Frankfurter Rundschau" newspaper for 1988 were examined. The corresponding text consists of 1,671,136 words, of which 77,745 are unique. The computer system and examined 5,849 possible error strings f_{ij}, of which 643 were actual error strings F_i. The rules R_j indicated in Table 1 to Table 19 were applied, whereby the application of rules R_2 and R_3 alone resulted in detecting 295 different actual error strings F_i.

[0100]

[Beneficial effect of the invention] According to the present invention, detection or correction of error strings in text can be performed.

[Brief description of the drawings]

[Figure 1] is an outline flow chart of a first embodiment of the present invention.

[Figure 2] is an outline flow chart of a second embodiment of the present invention.

[Figure 3] is a view showing a storage structure that is ideal for storing strings according to the present invention.

[Figure 4] is a view showing a computer system according to the present invention.

[Explanation of the reference symbols]

1 storage means

2 processing means

3 storage means

4 storage means

5 means

6 means

7 means

8 means

9 calculation means

10 means

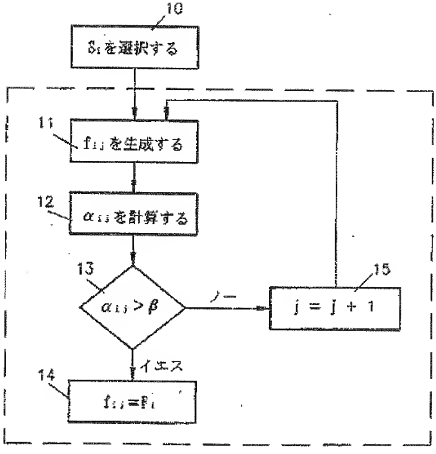
11 means

12 storage means

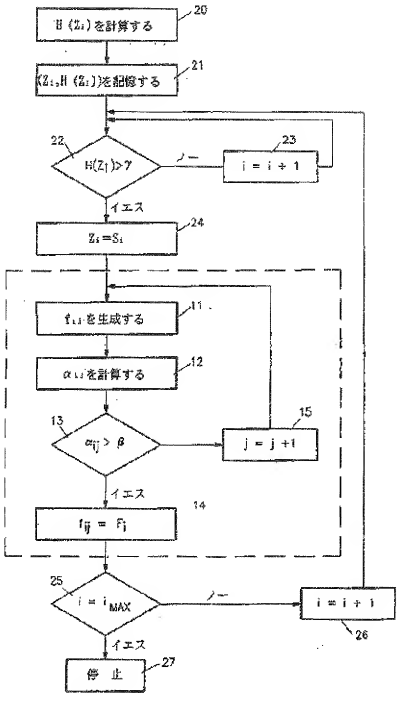
13 comparison means

16 bus

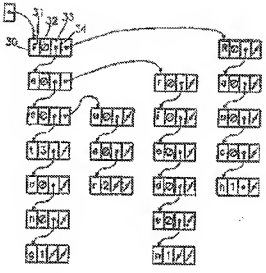
【図1】


10
S_iを選択する
11
f_{ij}を生成する
12
α_{ij}を計算する
13
$\alpha_{ij} > \beta$
ノー
15
j = j + 1
イエス
14
$f_{ij}=F_i$

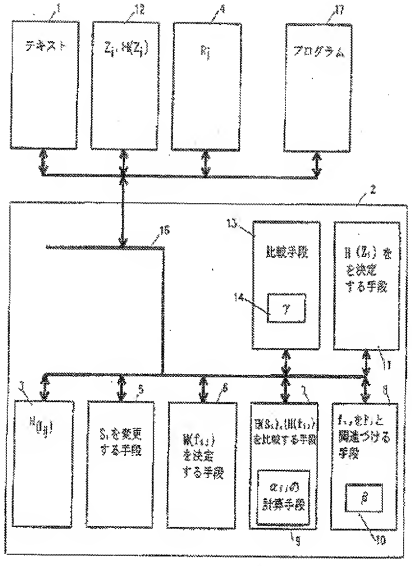
【図2】


H(Z_i)を計算する
20
{Z_i, H(Z_i)}を記憶する
21
22
H(Z_i)>γ
ノー
23
i = i + 1
イエス
Z_i = S_i
24
f_i,j を生成する
11
α_i,j を計算する
12
13
α_ij > β
15
j = j +1
イエス
f_ij = F_j
14
25
i = i_MAX
ノー
i = i + 1
26
イエス
停 止
27

【図3】



[図4]



[Figure 1]
10 selection of S_i
11 generation of f_{ij}
12 calculation of α_{ij}
[link between 13 and 15]: No
[link between 13 and 14]: Yes

[Figure 2]

20 calculation of $H(Z_i)$
21 calculation of (Z_i and $H(Z_i)$)
[link between 22 and 23]: No
[link between 22 and 24]: Yes
11 generation of f_{ij}
12 calculation of α_{ij}
[link between 13 and 14]: Yes
[link between 25 and 26]: No
[link between 25 and 27]: Yes
27 stop

[Figure 4]
1 text
5 means for modifying S_i
6 means for determining $H(f_{ij})$
7 means for comparing $H(S_i)$ and $H(f_{ij})$
α_{ij} calculation means
8 means for associating f_{ij} and F_i.
11 means for determining $H(Z_i)$
13 comparison means
17 program